设计实验室 5 6.01 –Fall 2011

惊人的比例

目标： 在本实验中，我们将对机器人进行编程，使其沿着墙壁移动至一侧，并保持与墙壁恒定且期望的距离。我们将使用一个简单的*比例控制器*，并将其建模为一个具有与**设计实验4**中用于*墙壁*探测系统相同的结构的系统。

这将分三个步骤进行研究：

- 为机器人构建一个比例控制器，并在模拟环境中对其进行测试不同的收益。
- 构建控制器 - 植物 - 传感器系统的分析模型，两者通过手动操作并使用 SystemFunction 类。
- 使用该模型来了解应为控制器使用何种最佳增益，以及应预期系统会出现何种行为。

1 引言

资源： 本实验应与搭档一起完成。每组搭档应有一台可靠的运行 SOAR 的实验笔记本电脑或个人笔记本电脑。

- propWallFollowBrainSkeleton.py：带有模板的大脑，其中有一个位置供您编写比例控制器。
- 设计实验室 05 工作.py：带有适当注释的 Python 模板脚本用于实现系统功能并查找其属性的端口。
- 课程笔记的第五**章**

一定要把你的所有代码和图表寄给你的搭档。你们每个人都需要在第一次面试时带上副本。

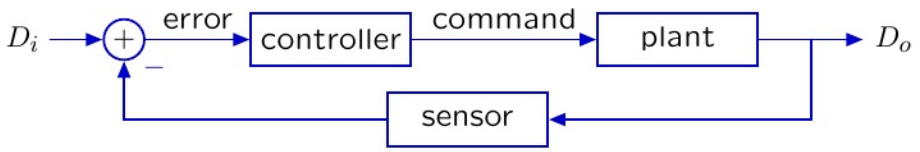
我们一直在研究一个基本的反馈系统，它由三个部分组成：控制器、植物和传感器。在上周的实验中，这个系统被建模为一个差分方程，其中输入是期望的距离到墙壁，输出是实际的距离到墙壁，而声纳提供反馈，关于感知的距离到墙壁。

本周，我们从简单的一维模型推广到二维世界，并试图引导机器人，使其与墙壁平行移动，并保持特定的距离。

Some of the software and design labs contain the command athrun 6.01 getFiles. Please disregard this instruction; the same files are available on the 6.01 OCW Scholar site as a .zip file, labeled Code for [Design or Software Lab number].

此外，我们从差分方程推广到使用抽象信号和系统方程的模型。

我们所构建的系统的基本结构将保持不变：



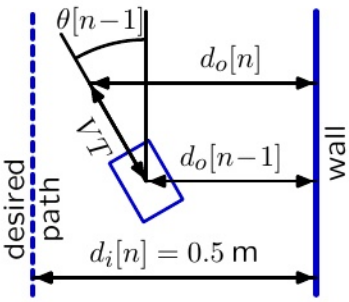
然而，与该图表相关的符号现在是信号，而不仅仅是样本。具体而言，以下是本次实验中使用的各种符号的定义：

- k：控制器的增益，一个常数
- V：机器人的前进速度，一个常数
- T：信号离散样本之间的时间间隔，一个常数
- D_i：期望的；机器人到墙壁的距离，一个其样本为$d_i[n]$的信号
- D_o：机器人到墙壁的实际距离，一个样本为$d_o[n]$ 的信号
- E：误差，等于 $D_i - D_o$，一个其样本为 e[n] 的信号
- Θ：机器人相对于墙壁的角度，其样本为 θ[n] 的信号
- Ω：机器人的角速度，其样本为 ω[n] 的信号

2 比例式壁面跟随器

目标： 为机器人构建一个比例控制器，并在模拟环境中用不同的增益对其进行测试。

右侧的图展示了走廊中一个机器人，其期望路径与右侧墙壁的距离为固定距离。我们可以构建一个控制器，其固定的前向速度为V = 0.1 m/s，并调整旋转速度ω[n]（未显示）与*误差*成正比，误差是期望距离$d_i[n]$ = 0.5 m到右侧墙壁与实际距离$d_o[n]$之间的差值。误差与旋转速度之间的比例常数称为*增益*，我们将它记为k。请注意，当机器人的旋转速度ω[n]为正时，机器人会向左转向，从而增加其角度θ[n]。



查看文件 propWallFollowBrainSkeleton.py。大脑有两个级联连接的状态机。第一个组件是 Sensor 类的一个实例，它实现了一个状态机，其输入类型为 **io.SensorInput**，输出为与右侧墙壁的垂直距离。垂直距离是通过 sonarDist 模块中的 getDistanceRight 函数使用三角测量计算得出的（假设墙壁在局部是直的）。提供了 Sensor 类的所有代码。

大脑的第二个组成部分是 WallFollower 类的一个实例。你的任务是提供代码，使 WallFollower 类实现一个比例控制器。

自我检查 1. 对于 WallFollower 类的状态机，其输入和输出应该是什么类型？k 应该是什么符号？

详细指导：

步骤 1. 通过编辑大脑来实现比例控制器。然后使用 SOAR 在 worldWallTestWorld.py 世界中运行您的大脑。该大脑的设置是在设置期间发出一个命令来旋转机器人，因此它以相对于墙壁的小角度开始。

步骤 2. 对增益 k 的几个值进行实验，以确定 k 如何影响所得的行为。

生成图表来说明您发现的趋势。

保存你的情节的**屏幕截图**，然后通过电子邮件发给你的搭档。

核对 1。 **第 5 周 3.1**：向一位工作人员展示你的图表。

描述增益 k 如何影响壁面跟随者的行为。

对于什么样的 k 值（如果有的话），到墙壁的距离会以最快的速度收敛到期望的 0.5 米值？

比较本次实验中 k 的效果与实验 4（找墙）中的效果。

3 数学模型

目标： 构建比例壁面跟随系统的解析数学模型。

现在我们将构建一个系统模型，以便能够从分析上确定系统的性能如何取决于增益。与使用 SOAR 试图详尽地搜索参数空间相比，**这种分析**方法更不繁琐且更具洞察力。

您可能会发现阅读阅读材料**中的第 5.8.3** 节会有所帮助，该节为寻墙系统说明了类似的建模工作。

第三步. 控制器模型：假设控制器能够立即将旋转速度ω[n]设定为与误差e[n]成正比。将这种关系表示为一个差分方程。

第四步. 植物模型：写出描述“植物”的差分方程，该方程将输入（角速度）与输出（到墙壁的距离）联系起来。将这个问题分为两部分是有用的:

- 方案 **1：**写出θ[n]的表达式，即机器人相对于墙壁的角度方向，它取决于其旋转速度。假设在时间 n - 1 时的旋转速度为ω[n - 1]，并且该旋转速度在时间 n 之前保持恒定。

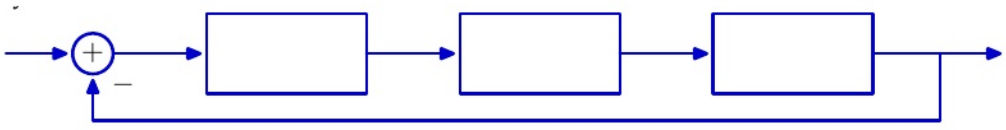
- **植物 2：**写出d_o[n]的表达式，该表达式取决于角位移。假设在时间 n - 1 时的角位移为θ[n - 1]，并且该角位移在时间 n 之前保持恒定。

  使用小角度近似（即，如果θ很小，那么sinθ≈θ）将所得差分方程线性化。这种近似使我们的模型线性化，便于分析。然而，当考虑这种近似的后果时，可能会有所帮助。

  试图在后续部分解释这些行为。

传感器模型：为了简单起见，我们将把传感器建模为一条电线：也就是说，假设它不引入延迟。

由上述三个差分方程表示的子系统相互连接，形成一个如下所示的系统。在方框图中，用相应信号的名称给每条电线标注。您可能还会发现，用与您推导出的三个差分方程相对应的模型元素（控制器、植物1、植物2）给此图中的每个方框标注也会很有用。



第 5 步：将差分方程转换为算子（R）方程，并求出系统函数。

$$H = \frac{D_o}{D_i} =$$

第 5 周 3.2 通过输入分子将系统功能输入到导师系统中。
分母多项式

第 6 步：为系统的极点 p 找到一个代数表达式。

p=

第 7 步。选择其中一个极点 p，并绘制两幅草图（在同一坐标轴上），分别展示当 k = 1、T = 0.1 秒且 V = 0.1 米/秒时，p^n的实部和虚部。该图向您揭示了关于**系统响应**的哪些信息?

核对 2。 **第 5 周 3.3** 节：描述在**第 6** 步中极点的位置。这些位置对系统行为有何影响? 将第 **7 步中的绘图与**检查 **1 的结果进行比较。解释相似之处**和不同之处。周期相同吗?

4 软件模型

目标： 使用 **SystemFunction 类**构建比例跟踪壁面系统的模型。

我们现在开发一种系统功能的软件表示法，以自动完成前面大部分的代数运算。该软件解决方案允许对大型系统进行分析，这些系统很难通过手工进行分析。然而，我们的软件解决方案需要诸如k和T等参数的数值。（您可以在更高级的编程课程中了解此类参数的符号处理系统。同时，与**信号**的**软件表示法**进行比较和对比。）

SystemFunction 类提供了两种基本的系统函数：增益（sf.Gain）和延迟（sf.R）。它们是通过 Python 程序实现的，但以大写变量命名的，这与 sm.Gain 和 sm.R 状态机类似。

```
def Gain(k):
    return sf.SystemFunction(poly.Polynomial([k]), poly.Polynomial([1]))
def R():
    return sf.SystemFunction(poly.Polynomial([1, 0]), poly.Polynomial([1]))
```

可以使用 sf.Cascade（级联）、sf.FeedbackSubtract（反馈减法）和 sf.FeedbackAdd（反馈加法）将这些组合起来，以创建任何可能的系统函数；并且组合的结构将与构建类似状态机的结构相同。

请注意，虽然增益（Gain）和电阻（R）的内部定义使用**多项式**来构建它们，但通过抽象，您可以使用这些系统函数以及我们的组合方法，而无需利用这些内部细节。

自我检查 2. 使用增益、延迟和加法器来绘制系统图，以表示上一节中的控制器、植物 1 和植物 2 。

控制器:

植物 1:

工厂 2：

第 8 **步**。编辑设计 Lab05Work.py 文件，以实现名为 controller（控制器）、plant1（植物 1）和 plant2（植物 2）的 Python 程序，这些程序使用 sf.Gain（增益）、sf.R（电阻）、sf.Cascade（级联）、sf.FeedbackSubtract（反馈减法）和 sf.FeedbackAdd（反馈加法）来构建并返回 SystemFunction 类的实例，这些实例代表数学模型中的三个模块。将每个模块的重要参数（例如 k）作为输入传递给相应的程序。

第 9 步。编写一个 Python 程序 wallFollowerModel（k， T， V），调用前面的 Python 程序为组件创建系统函数，并将它们组合成一个单一的系统函数，该系统函数以 desiredRight 作为输入，以 distanceRight 作为输出来描述系统。

第 5 周 3.4 节 将 wallFollowerModel 的定义以及它调用的任何程序输入到导师系统中。不要输入任何导入语句。

第 10 步。使用您系统模型的主导极点法来确定当 T = 0.1 秒、V = 0.1 米/秒以及您在 Checkoff **1** 中使用的 k 值时所产生的周期。

k 时期；时间段

检查 3. **第 5 周 3.5**：向一位工作人员展示您的成果。解释您在本部分的结果与**检查 1** 的结果之间的异同。